

REMARKS

Claims 125-148 and 153-182 remain in the application. Reconsideration of the application and allowance of all claims are respectfully requested in view of the above amendments and the following remarks.

All claims stand rejected under 35 U.S.C. 103(a) as unpatentable over "RPC over HTTP via XML" (hereafter "Winer") in view of "W3C XML". This rejection is respectfully traversed.

As discussed in the Amendment filed May 7, 2003, a central aspect of the present invention is the inclusion in the message of type labels in association with data items which are either themselves arguments or are contained within arguments. A further and critical point to note is that the term "type label" has a clearly defined meaning in the present application. Lines 6-14 of page 4 of the specification explain that a type label, as that term is used in the present application, refers to a label which identifies a programming language data type, e.g., integer, float, long, string, Boolean, as well as programming language data structures such as array, record, and vector. There are other examples given elsewhere in the specification. What is important to note is that these type labels are included in the message and tell the receiving application what it needs to know to interpret the data, so that the sending and receiving applications do not have to agree in advance on a particular type of data or a particular order of data items.

Finally, and very importantly, the invention defined in the present claims is not simply the use of XML for RPC, or even the indication of types for arguments. The invention resides in a particular manner of designating those types, in a way that is not shown in the prior art and would not have been obvious from that prior art.

Turning in more detail to the specific references relied on in the rejection, the Winer publication is no more pertinent to the presently claimed invention than the earlier-discussed NC.Focus paper or other art of record broadly suggesting the use of XML for RPC. There is nothing but the most general of descriptions of using XML to implement RPC. The section beginning at page 5 of the paper is the only part of the paper that mentions XML RPC, and only the last two paragraphs of that section say anything about XML RPC:

This is important because there is another layer coming on the Internet, a very simple one, that can build on COM, on Windows and elsewhere, and provide a flat playing field for everyone.

It's RPC over HTTP via XML. I believe it's the next protocol for runtimes. We'll be posting more notes on this on Scripting News over the next few weeks. Please, if you care about the independence of the Internet, tune in. Thanks!

Winer does not even begin to teach anything about the use of type labels in association with data items of arguments, much less the specific combinations of type and semantic labels disclosed and claimed in the present application. In his remarks at page 3 of the Office action, the examiner alleges that Winer teaches (1) a message which includes plural elements representing data items of at least one argument, and (2) association of the data items with type labels selected from an encoding group. But in the very brief and completely non-technical discussion of XML RPC, Winer never refers to any data items or arguments, and certainly does not say anything about providing type labels for them. And it is certainly not inherent in the use of XML for RPC that type labels will be provided for the arguments, since it is at least possible to do it some other way. If the examiner continues to assert that Winer teaches the association of the data items with type labels, the examiner is respectfully requested to point out more particularly where such teaching can be found, so that applicants can respond.

The secondary reference, W3C XML, is a draft specification for an XML standard for describing documents. XML is a markup language to be used for forming markup documents containing character data and also containing markup. The element labels that XML provides are not suitable for the purpose of the present invention. They are designed for documents, and they convey information about what a document *looks* like, but not what it *means*. If one were to simply follow the lead of the XML specification, one would use element labels to show what the RPC XML "document" looks like, but not what it means. But that would not be enough for RPC, where the receiving station must be able to determine how to interpret the RPC elements, and not merely know what they look like.

For example, with reference to page 14 of W3C XML, Section 3.1, that specification describes the use of start- and end-tags for each element which include a *Name* which designates

the element type. But the "element type" referred to here is not a data type of the sort that is defined by a programming language, i.e., the W3C XML element type does not tell the receiving system how to interpret the data contained within the element but is instead perhaps more analogous to the semantic labels used in the present invention. No specific examples are given in the W3C XML paper, but in the later version (W3C Recommendation 10-Feb-98, cited by the examiner as Reference W in the Office action of August 5, 2003) at the corresponding section 3.1 on page 14, an example of a start tag is given where the *Name* included in the start tag is "termdef". There is no suggestion anywhere in either reference that "termdef" is interpreted as designating a particular programming language data type. Indeed, as noted in the second full paragraph of the section "3. Logical Structure" at the top of page 14 of W3C XML, there are no constraints on the use of names of elements. In other words, there is no limit on the number of names that can be used, as long as each name is declared in the DTD. So the element names in W3C XML cannot possibly correspond to the claimed encoding group of element type labels. There are an unlimited number of names that can be used, and these would identify data and allow retrieval of data according to such identifications. This characteristic of XML is discussed in the present application at pages 15-18. But the invention defined in the present claims relates to the use of type labels to designate a defined group of element types.

Since XML allows an unlimited number of names, then it is not possible to have the names themselves designate an element type. This is in contrast to the present invention wherein data items of elements are associated with type labels belonging to an encoding group, with each label designating a programming language type, e.g., the examples given at pages 30-31 where all of the data items are labeled "VALUE" in addition to having semantic names, or the example at pages 47-48 where each data item begins with a label designating its type. There is no suggestion anywhere in W3C XML that this type of labeling of the data items would be advantageous.

The closest thing to the claimed invention that can be found in XML is the "DTD." The mixed content and element content types (discussed in more detail below) are something that the W3C XML publication teaches as occurring in an Element Declaration (see, e.g., Section 3.3 at

pages 15-16), the purpose of which is described as establishing a set of structural constraints, i.e., a grammar, for the document. It is in this Element Declaration that the "type" (as that term is used in the W3C XML specification, but not synonymous with the same term in the present application) of an element would be defined using the mixed content indicator. This Element Declaration will occur as part of the document type declaration (see, e.g., Section 2.8) which occurs in a prolog of the XML document and will typically refer to both internal and external subsets of markup declarations, collectively known as the document type definition (DTD).

The DTD is a schema definition language for XML. It labels data according to whether it can hold mixed content, and can define the content of specific structures, but a DTD does not define programming language data types or structures. These DTD-defined types are then used in the XML document to label data items. What is critical is that the *meaning* of these types is not given by the DTD. The DTD could be defining semantic labels or data type labels or some other kind of label -- that's up to the application that interprets the XML. The 80/20 XML-RPC described in the present specification is an application that defines **specific** types, types that are found in programming languages. It says that if you use certain names, whether in the DTD or otherwise, then these names shall represent programming language types, which we call "data types" in the specification.

So in a broad sense, each tag in an XML document may signify the "type" of data that the tag is attached to, but XML does not say that these are "data types" as that term is used in the present application, and it does not standardize which tags correspond to which data types. That standardization is one of the features to which the present claims are directed.

Since the examiner has at many points referred to the "mixed content" designation in W3C XML, it bears noting at this point that "mixed content" does not describe a type of the sort required of the present application. Mixed content is described in W3C XML as indicating that an element contains character data optionally interspersed with child elements. As noted at page 17 of the present application, this would not be particularly useful when representing data. Further, it does not tell the receiving system the data type of the character data. According to the present invention, the sending and receiving ends could agree that the label "termdef" designates a string data type. But there is no suggestion anywhere in the publications relied on by the

examiner that such an agreement would have been advantageous, and indeed it is not surprising that there is no such suggestion because the W3C XML publications are describing a language that was at that time contemplated as being useful to describe the **appearance** of documents, not how any of the data is to be interpreted. Thus, it is clear that when the W3C XML paper is referring to element types, it is not referring to a type as that term is used in claims of the present application.

If one of skill in the art were to have considered together the teachings of Winer and W3C XML, applicants do not contest that Winer would have suggested to the artisan that XML could be used to implement RPC. Learning then about XML and reviewing the W3C XML specification, the artisan would have seen the use of start- and end-tags for each element, the use of element *Names* to perhaps indicate the role of the element in the syntax of the XML document, and in some cases something about the structure or appearance of XML document. The artisan may have designed the XML RPC encoding to do these same things. But this would not have resulted in the claimed invention. More inventive work would have been required, since there is nothing in either of these documents to have suggested using the specific kinds of type and semantic labels recited in the present claims.

As explained earlier, conventional RPC requires that the recipient of a message know in advance the data types of the message arguments found in the message. Conventional RPC extracts the data items from a message according to the order in which they appear in the message, and it can do so only because it knows in advance what data types appear at what positions in the message. If one were to use XML to implement conventional RPC, one might have, e.g., a <MESSAGE> element for the message and an <ARGUMENT> element for each input or output argument of the service invoked. Neither MESSAGE nor ARGUMENT provides a type label as claimed. These labels may tell the receiving system the significance of an element, e.g., it is an input or output argument. But the receiving system still cannot interpret the data because it knows nothing about the "type" of the data. It would be different if the receiving system knew to assume that unless otherwise indicated a data item found within an <ARGUMENT> element is of string type, but that is not what the W3C XML reference is about.

To interpret this data, a recipient must still, e.g., know in advance what data types occur at what positions. Further, the infrastructure for conventional RPC is typically geared toward sequentially serializing message arguments on outbound requests and toward sequentially deserializing message arguments on inbound replies. So even if one thought to represent an RPC in XML, it would not have been clear how one would fit this into existing RPC-middleware. Indeed, the NC.Focus paper, cited earlier, discusses building hierarchical structures of name/value pairs using XML, but still requires knowledge of the context because there are no type labels associated with arguments.

Discussion of Specific Claims

Claim 125 requires that the service invocation request include plural elements representing data items, that each data item is associated with a type label (as that term is defined in the specification) and that the type labels are selected from a group including at least two members (e.g., the RECORD, LIST or ARRAY labels discussed at pages 43-45 of the present application) designating elements containing other elements associated with type labels belonging to the group. This is not found in either cited reference. The examiner characterizes Winer as teaching a message encoding where data items of an argument are associated with type labels, but no such teaching can be found anywhere in Winer. There is simply a broad suggestion of using XML for RPC, and no discussion whatsoever as to designating the programming language types of data items of arguments. More importantly, there is no discussion of *how* these types would be designated. As noted above, if the examiner will continue to rely on Winer for a teaching of associating data items of arguments with type labels designating programming language types, he is requested to explain in more detail where such teaching can be found so that applicants can properly respond.

The examiner then relies on W3C XML to teach an encoding group that includes two members designating elements containing other elements. The examiner has not identified a group of indicators of programming language types, much less identified where in W3C XML there is a teaching of two elements of such a group having the characteristics recited in claim 125. The examiner refers applicants' attention to "3.3.1 Mixed Content" discussion and

associated declaration at page 16, but this does not teach even a single type label as that term is defined in the present application, much less plural such type labels which are part of a defined encoding group. The mixed content type is in fact already described at page 17 of the present application, and will be discussed again in more detail below.

First, and importantly, *mixed content* does not describe a type of the sort required of the present application. *Mixed content* is described in W3C XML as indicating that an element contains character data optionally interspersed with child elements. As noted at page 17 of the present application, this would not be particularly useful when representing data in a programming environment such as RPC. Further, it does not tell the receiving system the data type of the character data.

Thus, even if the teachings of W3C XML were followed, there is nothing to suggest labeling each data item with a label selected from an encoding group and which designates a programming language type. The examiner has not identified the "encoding group" in the W3C XML publication, nor has the examiner pointed to any teaching in W3C XML (or in Winer) of a label which itself designates the element type. It is respectfully submitted that there is no such teaching in the prior art, and that the subject matter defined in claim 125 would not have been obvious to one of ordinary skill in the art from the teachings of Winer and the W3C XML publication.

The basic point of distinction discussed above with respect to claim 125 applies to all claims presently pending in the application, i.e., they all require labeling of elements with type labels selected from a group and which designate programming language types. This is simply not suggested anywhere in the prior art. Beyond this, there are distinctions in either the particular kinds and combinations of labels, or in the manner in which they are provided, which are reflected in the language of various claims, and these will be discussed below.

All of claims 125, 126, 131-137 and 141-144 require the provision of two labels each designating elements as containing other elements, and with all of the contained elements being associated with type labels belonging to the encoding group. This is not the case in W3C XML.

Claims 127-130 further require that the encoding group of type labels include a first label (e.g., the VALUE label discussed at page 43 of the present application) designating an element containing lexical data and a second label (e.g., the RECORD label of the present application) designating an element as containing other elements associated with labels from the encoding group. In his remarks in support of the rejection at pages 4-5 of the Office action, the examiner refers to XML element declarations where *Name* is allegedly a type label indicating an element containing lexical data, but this is not so. The *Name* indicates a functional type within the XML syntax, but does not designate that the element contains lexical data. The receiving station requires further information to know the data type of whatever is contained in the element. As discussed above, there is no limit on the number of names that can be used, and not all named elements contain lexical data, so there must be some mechanism other than the *Name* label alone by which the receiving application will know that an element in fact contains lexical data. W3C XML does not explain how this is accomplished, and does not teach the claimed invention. Further, there is nothing in W3C XML to teach that all elements contained in a container element will be associated with type labels from the encoding group, since W3C XML does not teach such labeling as discussed above.

Claim 131 in combination with its parent claims requires that the available type labels include three different type labels, with two designating container elements containing elements associated with type labels designating programming language types and a third type label designating an element containing lexical data. The examiner has simply referred to the mixed content discussion in the W3C XML publication, but this does not support the recitation of three different type labels as required of claim 131.

Claim 132 depends from claim 131 and in combination therewith requires that the available element type labels include not only the three required by claim 131 but also a fourth label designating a container element, i.e., now requiring the existence of three different labels (e.g., RECORD, ARRAY, LIST) all designating container elements. The examiner simply refers again to the mixed content discussion in the W3C XML publication, but this cannot reasonably be considered to support the teaching of the four type labels required of claim 132.

Claim 133 requires, in addition to the lexical type label and two container type labels recited in claim 131, a further type label designating another element which uniquely identifies another element within the message. This refers to the OBJECT type label discussed, e.g., at page 44 of the present specification. The examiner again refers to the mixed content discussion of the W3C XML publication, but this does not discuss a type label of the kind required in claim 133, or the three additional type labels required in the parent claim of claim 133. The most relevant discussion in the W3C XML publication is the discussion of the use of ID and IDREF attributes by which an element can be used to refer to another element, but this would still not supply the teaching missing relative to the other labels required of the claim, as discussed above.

Claims 145-148 all refer to a type label designating an array element. W3C XML does not teach an array type label, and there is certainly no such teaching to be gleaned from simply the "mixed content" feature of W3C XML. But even further, the claim requires that there be at least one array element representing a multi-level nested array where each element nesting level corresponds to a respective dimension of the array. This concept is simply nowhere reflected in W3C XML.

Claims 153-156 require a type label designating an n-dimensional array (where n is greater than or equal to 2). The examiner has once again referred to the mixed content feature of W3C XML, but this does not relate at all to multi-dimensional arrays. There is also no suggestion anywhere of having an array and also of having an array label (e.g., the optional TYPE attribute described at lines 3-12 of page 45) that requires that all children of the array be required to have the same type as one another. Thus, the subject matter of claims 153-156 is not taught in the applied art.

Claims 165-168 describe messages having first and second elements having different types, and having one of the elements associated (e.g., by means of the ID attribute discussed at lines 20-26 of page 45) with an ID value and the other element specifies the value of the ID. In support of his rejection of this claim, the examiner refers to the discussion of tags beginning at page 14 of the W3C XML publication. However, this discussion merely relates to start and end tags having attributes. Of more relevance is the discussion of ID and IDREF attributes at page

18. However, this subject matter of claims 165-168 is not simply the use of objects and references, but the manner in which this is done. The present invention as defined in claim 165 provides a set of type labels and labels the data items. As discussed above, W3C XML does not teach the use of type labels as that term is defined in the present specification. In addition to the use of such labels, claim 165 describes the association of an element having a first type label with an ID value, and the inclusion of an element specifying said ID value and associated with a second type label. In the context of the disclosed invention, the element associated with a "first" type label may be any element in the encoding, as noted at lines 11-12 of page 44. The "second" type label is the OBJECT label which has a mandatory REFERENCE attribute. W3C XML does not suggest the use of a type label such as OBJECT which always refers to a data item having a mandatory IDREF attribute. It is believed that claims 165-168 already clearly define the distinction, but these claims have been further amended to make the distinction even more clear. This subject matter is simply not taught in W3C XML, where not only are type labels not used but there is no type label that has a mandatory REF attribute.

Claims 169-172 describe having a placeholder type label designating a placeholder element representing the absence of data. In support of his rejection of these claims, the examiner refers to the discussion of empty element tags in the W3C XML publication. It is noted that the W3C XML publication provides for the designation of an empty element either by a start tag followed immediately by an end tag, or by a start tag having a special form. The special form of the start tag involves the use of a forward slash "/" at the end of the start tag. It does not rely on the use of an element type label as is described in claims 169-172. An alternative in W3C XML is for the element to have been declared as being empty, but this is not done by type labeling.

Claim 134 recites the use of a placeholder element type label and is similar in that regard to claims 169-172 discussed above, but this recitation in dependent claim 134 comes in the context of the earlier recitation of having a lexical type label and at least two container element type labels. This combination of labels is simply not suggested in W3C XML.

Claim 135 recites a placeholder element type label (e.g., OBJECT) and is therefore patentable for the same reasons as discussed above with respect to claims 169-172, and further recites the use of an element type label (e.g., NULL) for designating the absence of data, and is therefore patentable due to the absence of a teaching in the prior art to include type labels for designating both of these.

Claim 136 refers to a container element type label but in combination with its parent claims now requires a set of labels including three different container labels (e.g., ARRAY, RECORD, LIST), one lexical label (e.g., VALUE), one referencing label (e.g., OBJECT), and an absence label (e.g., NULL). The result is a set of element type labels that permits efficient representation of the arguments in an RPC in the manner discussed in detail at pages 41-45 of the specification.

Claim 137 is dependent on claim 131 and further specifies that one of the container type labels actually designates a multi-dimensional array. This refers to the use of a label (e.g., the ARRAY type label of the present invention) for designating an n-dimensional array. The examiner refers to the "mixed content" feature of W3C XML, but there is nothing to suggest that mixed content is synonymous with an n-dimensional array.

Claim 138 reflects the use of a lexical type attribute in association with an element that has been associated with an element type label indicating lexical data. W3C XML discusses the use of attributes, including a string type attribute, but not in conjunction with an element that has already been designated by its element type label as containing lexical data. Thus, the subject matter of claim 138 would not result from the combination of the teachings of the applied art.

Claim 139 reflects the feature of the present invention whereby an element that has been designated by its type label as being of lexical type will be assumed to be of string lexical type if there is no attribute that specifies a different lexical type. The examiner refers to the phrase from page 9 of W3C XML which refers to non-textual data as by definition containing no elements, but this phrase has nothing to do with the presence or absence of a lexical type indicator, and clearly does not teach or suggest the subject matter of claim 139. W3C XML, e.g., in Section 3.4.2, does discuss the use of declarations to establish attribute defaults. But declarations

establish defaults for the entire XML document, not defaults on an element type by element type basis. Further, since there is no element type in W3C XML that designates an element containing only lexical data, one cannot have a default type of lexical data keyed only to the element type. Thus, the subject matter of claim 139 would not result from the combined teachings of the applied art.

Claim 140 together with its parent claims requires a message that is generated using an XML-based message encoding with the type labels expressed as XML element type names, i.e., there must be an XML element type name designating a container element and an XML element type name designating that the element contains lexical data, and there must also be an XML attribute associated with the lexical element and having a value indicating the type of lexical data. This is simply not found in the applied art, where there is no element type name that requires that the only content of the element is lexical data. Even if the W3C XML publication were considered to teach the use of type labels in an RPC message encoding, it would not use the XML element type names themselves to designate programming types, but would at best do this in a DTD. The use of an XML element type name for this purpose, as in the present invention, results in significant simplification of the message encoding.]

Accordingly, the subject matter of claim 140 would not result from any obvious combination of the teachings of the applied art.

Claim 141 in combination with its parent claims requires a type label designating the element as containing lexical data, two different type labels designating their respective elements as containing other elements and a further type label (e.g., NUMBER, in the variation described at page 52) designating the element as containing a numeric value. W3C XML has or suggests no element type name designating a numeric element. While applicants have discovered this to be particularly useful in some implementations of RPC, it is certainly not suggested in the applied art.

Claim 142 in combination with its parent claims requires a type label designating the element as containing lexical data, two different type labels designating their respective elements

as containing other elements and a plurality of type labels each designating a different lexical data type (as discussed at lines 26-27 of page 52). This is simply not shown in W3C XML.

Claim 143 not only requires an element type label designating lexical data and two element type labels designating container elements, but also requires semantic labeling of at least one data item of an argument in the message. W3C XML arguably includes semantic labeling, but the entire premise of the rejection by the examiner seems to be that the labeling of W3C XML, when XML is used for RPC, would be used for type labeling. There is nothing to suggest that an encoded RPC include both the type labeling as required in claim 131 and the semantic labeling further required in claim 143.

Claim 144 is dependent on claim 143 and further requires that the semantic label be expressed as the value of an XML attribute. Again, this is not taught in W3C XML.

Claim 157 further requires that the type of all data items contained within an array type element may be designated by a single array label identifying that type. This is not suggested in W3C XML where there are no arrays discussed in any event.

Claim 158 further specifies that the array label is expressed as an XML attribute and specifies the dimension of the array. W3C XML does not discuss arrays, so it obviously does not suggest anything about a specific technique for indicating the dimension of an array.

Claim 159 further specifies that the message is an XML document. For purposes of the present rejections stated, applicants will rely only on the patentability of the parent claims.

Claim 160 refers to the use of type labels at each nesting level to designate the nesting element as having an array type. Again, with no discussion of arrays in W3C XML, this feature of the invention could not possibly result from any obvious combination of the teachings of the applied art.

Claim 161 is patentable due to the patentability of its parent claims, but requires that at some level where the nesting element contains a data item, all direct children of that nesting element have the same element type as one another. Again, there is simply nothing in the applied art that even touches on this subject matter, and it certainly could not be considered to

have been obvious from the applied art. The examiner refers to Section 3.3.1 of W3C XML, and this does allow XML to constrain the type of element which may appear as children. But there is nothing here about an array.

Claim 162 requires that the nesting element of claim 161 having a data item is associated with an array label which indicates the common type of all of the data items of direct children elements. Again, there is no teaching of labels of this kind or use in this manner.

Claim 163 refers to an array label which indicates the number of dimensions of the array but does not indicate the size of each dimension. The undersigned is unable to find anything in W3C having to do with arrays.

Claim 164 refers to the use of an array label to constrain the type of data appearing throughout an entire array, and not just within a single element within the array. This is not described anywhere in W3C XML.

Claim 173 requires that the placeholder element be a programming language null object reference. The examiner has simply referred to the empty element tag discussion in W3C XML, but there is no discussion there of programming languages at all, much less a null object reference.

Claim 174 requires that the placeholder element refer to an element contained elsewhere in the message. The only relevant discussion on this point in W3C XML would be the use of the IDREF attribute. However, this differs from the claimed subject matter for reasons given in detail above in the context of claims 165-168.

Claims 176 and 177 describe the association of a semantic label with the placeholder element. XML does provide for the use of semantic labels, but not for labels designating programming language types. If one were to use the XML semantic labels for programming language, one would still not end up with what is claimed here. The use of both programming language type labels and semantic labels is a significant improvement simply nowhere suggested in the art.

Claim 180 is dependent on claim 164 dealing with arrays, and specifies that the array label not only require that all data items within the array have the same type, but it also identifies what that type is. This is not found anywhere in the cited art.

Claims 181 and 182 require that all elements designating or representing data items be associated with type labels. Again, no individual labeling of data items with type labels is taught by W3C XML, much less the labeling of all data items.

At page 25 of the Office action, the examiner has rejected claims 128-148 and 153-182 for anticipation by Winer. This rejection requires little further discussion, in that detailed discussion is provided above as to why the subject matter of these claims is not taught by the combination of Winer and W3C XML, which inherently also must lead to the conclusion that the subject matter of the claims is not taught by Winer itself. Winer teaches nothing whatsoever about the features carefully recited in the present claims. Further, these features are clearly not inherent in XML. Withdrawal of this rejection is respectfully requested.

At page 26, the examiner has rejected all of claims 128-148 and 153-182 for anticipation by St Laurent et al. This rejection is respectfully traversed. First of all, even if such a public essay exists, the simple fact that it promotes XML RPC would make it no more relevant than the Winer paper already relied on. Second, and importantly, the St Laurent publication is effective as prior art only as of its own publication date and only for what it teaches itself, not for what the examiner may **suspect** that some other document which **may** exist **might** suggest. This rejection is clearly improper and withdrawal is requested.

At pages 27-28 of the Office action, the examiner has rejected claims 128-148 and 153-182 under the first paragraph of 35 USC 112. These rejections are respectfully traversed. The fact that claimed subject matter may or may not be supported in a provisional application is irrelevant to a Section 112 rejection. Measured against the disclosure of the present application, the claimed subject matter is clearly supported, and there is no basis for a section 112 rejection. The failure to find support in a provisional is only relevant to whether or not a claim is entitled to

the benefit if the provisional filing date. That issue is not at present relevant to any of the rejections stated, because applicants have not attempted to rely on the provisional filing date to overcome any of the stated rejections.

Summary

The examiner has relied on two references in the rejections discussed above. Winer is the primary reference and is relevant for nothing other than its general suggestion of using XML to implement RPC. W3C XML is an XML specification for describing documents. One feature common all claims is the use of labels for individual data items to designate programming language types. W3C XML does not do this. Another distinctive aspect is the use of particular combinations of element type labels to designate programming language data types. On this issue, neither of the references teaches how one would specify the kinds of things applicants have expressed using their element type names. Some of the things applicants have expressed using element type names are analogous to things in the XML specification but implemented in a different way, e.g., the use of a special start tag format instead of an element type name designating the absence of data, or the use of an IDREF attribute to indicate that an element references another element, rather than the use of an element type label (e.g., OBJECT) to designate an element which must have this attribute. In this connection, it is submitted that the use of element names to perform these designations is a significant simplification. In other aspects, e.g., the designation of arrays and the dimensions of arrays, the present invention is able to do this simply by element type labels and, e.g., nesting levels. The art cited by the examiner discusses nothing relating to such designations.

Further, the claims are not simply directed to using element type labels but to specific combinations of element type labels that applicants have found particularly advantageous in implementing RPC. Even if the art were considered to teach the use of element type labels generally in the manner of the present invention, the specific combinations are simply not taught. These might appear obvious through hindsight, but there is certainly no teaching in the art to support a proper obviousness rejection, and applicants' own inventiveness cannot be used against them. The examiner has frequently relied on a single concept (Mixed Content) to

simultaneously satisfy the requirement for multiple different type labels. But this is simply not supportable.

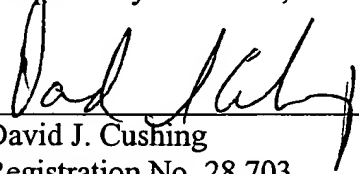
Beyond the specific combinations of element type labels used, the present invention provides significant improvement in its use of semantic labels in combination with element type labels. The semantic labels in the present invention are implemented as XML attributes. The prior art, particularly W3C XML, does teach the use of attributes, but does not teach both programming type and semantic labeling of data items. It teaches the use of element names but not for designating programming language types, and particularly not in the specific and advantageous combinations of type labels recited in the claims. The present inventors have found that significant advantages are realized by using element type labels and then using XML attributes to implement semantic labels, and this particular technique is simply not suggested in the art.

Further examination is respectfully requested.

A two-month extension of time is requested for responding to the Office action, and authorization is given to charge the extension fee to Deposit Account 19-4880.

SUGHRUE MION, PLLC
2100 Pennsylvania Avenue, N.W.
Washington, D.C. 20037-3213
Telephone: (202) 293-7060
Facsimile: (202) 293-7860
Date: January 5, 2004

Respectfully submitted,


David J. Cushing
Registration No. 28,703